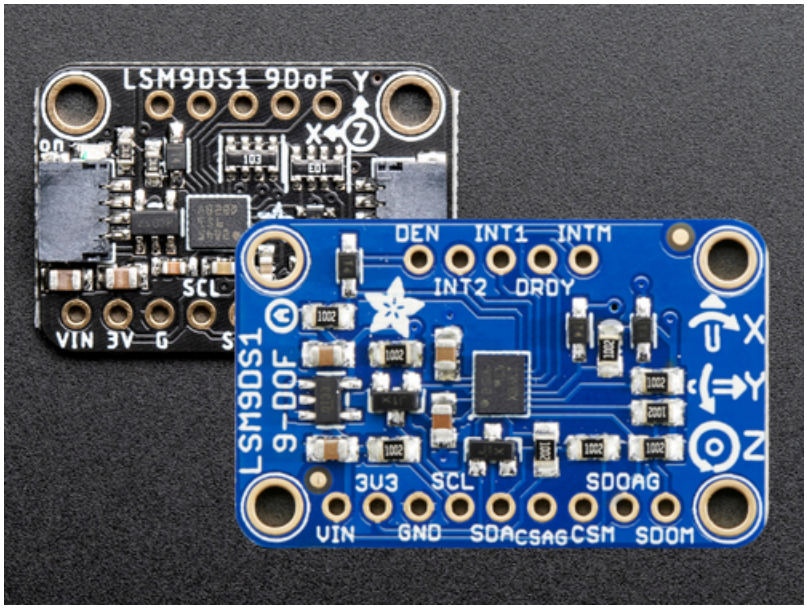
adafruit learning system

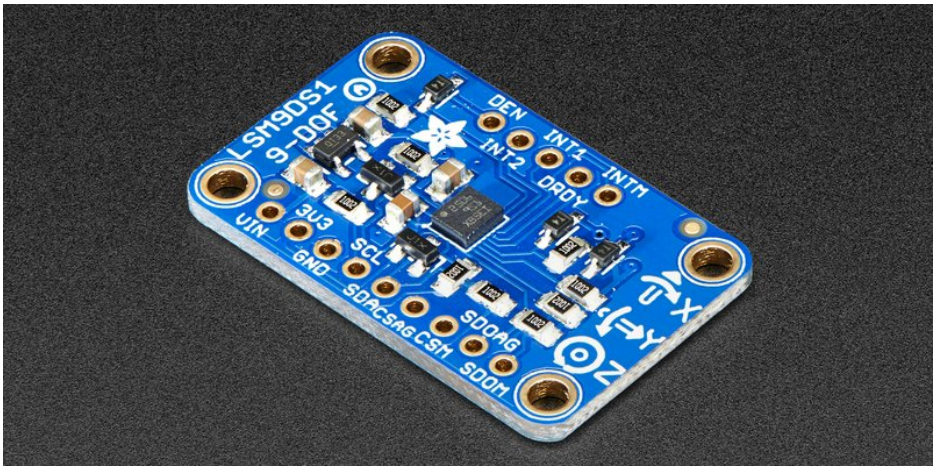
Adafruit LSM9DS1 Accelerometer + Gyro + Magnetometer 9-DOF Breakout

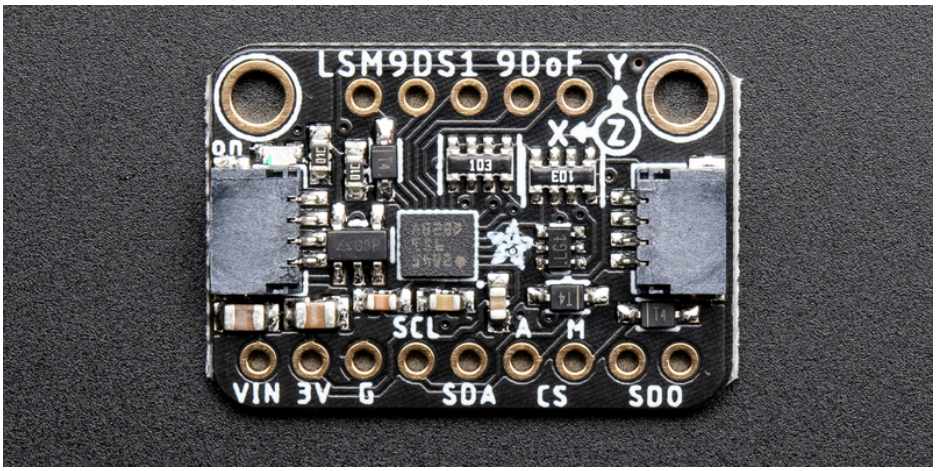
Created by lady ada



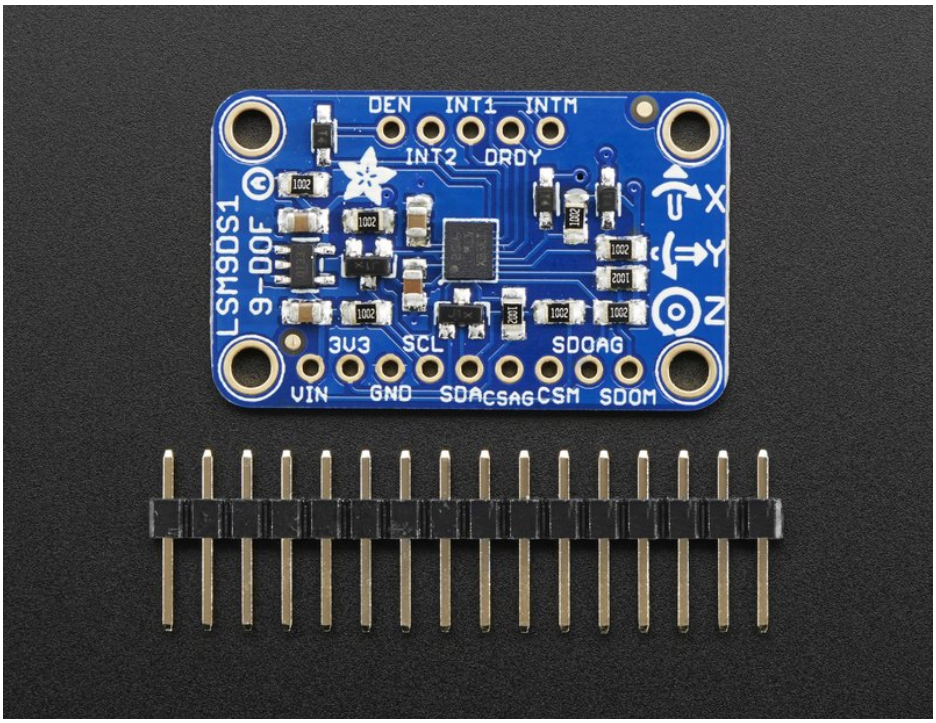
Last updated on 2020-06-25 12:29:17 PM EDT

Overview





Add motion, direction and orientation sensing to your Arduino project with this all-in-one 9-DOF sensor. Inside the chip are three sensors, one is a classic 3-axis accelerometer, which can tell you which direction is down towards the Earth (by measuring gravity) or how fast the board is accelerating in 3D space. The other is a 3-axis magnetometer that can sense where the strongest magnetic force is coming from, generally used to detect magnetic north. The third is a 3-axis gyroscope that can measure spin and twist. By combining this data you can REALLY orient yourself.



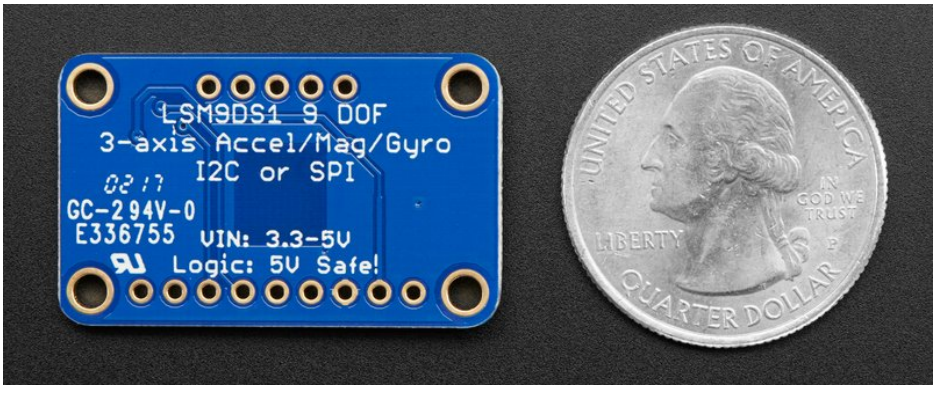
We've carried the LSM9DS0 from ST for a while, and the LSM9DS1 is their latest offering. We thought this could really make for a great breakout, at a very nice price! Design your own activity or motion tracker with all the data... We spun up a breakout board that has all the extra circuitry you'll want, for use with an Arduino (or other microcontroller)

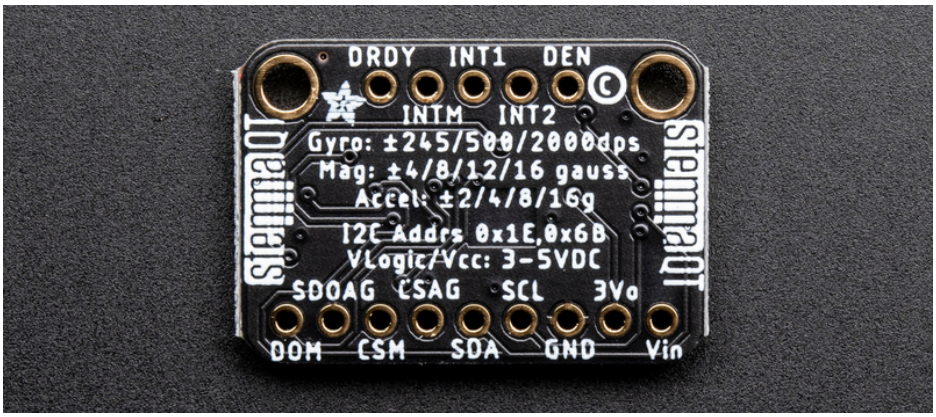
The LSM9DS1 is not the same set of sensors as the LSM9DS0. Here are some of the differences

- LSM9DS0 accelerometer has ±2/±4/±6/±8/±16 *g* ranges. The LSM9DS1 has ±2/±4/±8/±16 *g* (no ±6 *g* range)
- LSM9DS0 magnetometer has ±2/±4/±8/±12 gauss ranges. The LSM9DS1 has ±4/±8/±12/±16 gauss ranges. So the LSM9DS0 has ±2 gauss low range where-as the LSM9DS1 has ±16 gauss high range
- LSM9DS0 and LSM9DS1 gyros *both* have the same ±245/±500/±2000 dps ranges.

There are other differences, for example we noticed the LSM9DS1 has slightly worse accuracy. The gyro angular zero-rate (±25 for the LSM9DS0 and ±30 for the LSM9DS1 at the highest sensing range). The accelerometer offset accuracy is ±90 m*g* for the LSM9DS1 and ±60 m*g* for the LSM9DS0.

However, these offsets may not matter for most projects and the pricing of the LSM9DS1 is lower than the LSM9DS0



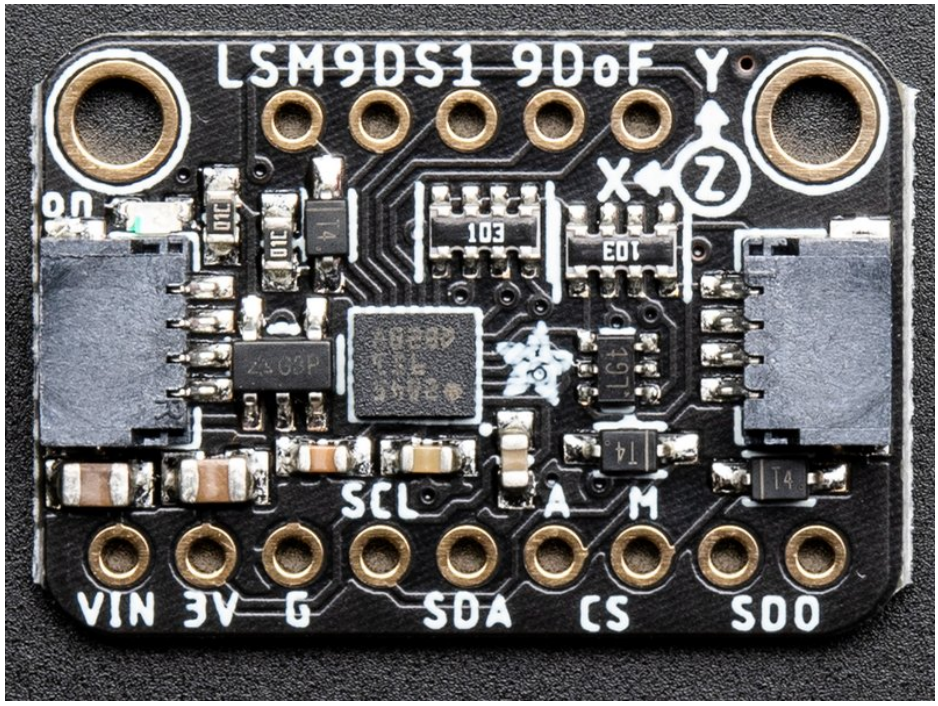


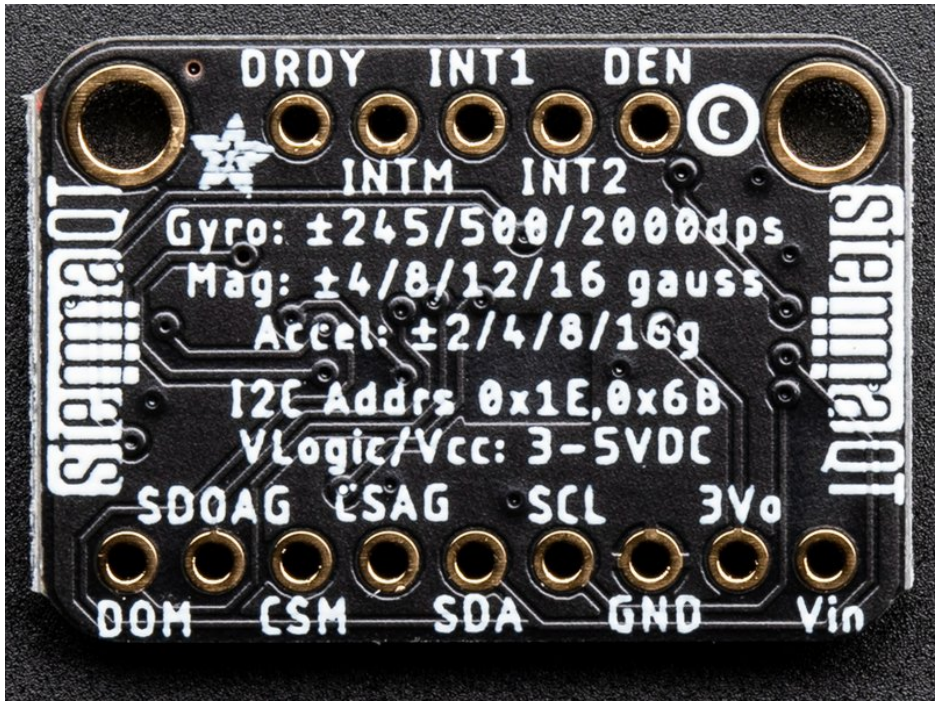
The breakout board version of this sensor has both I2C and SPI interfaces. Attaching it to the Arduino is simple, power Vin and GND with 3-5VDC, and wire up I2C data on SCL and SDA, and you're ready to go! More advanced users can use SPI, our library has support for both. The breakout comes fully assembled and tested, with some extra header so you can use it on a breadboard. Four mounting holes make for a secure connection, and we put the popular power+data pins on one side, and the interrupt pins on the other side for a nice & compact breakout.

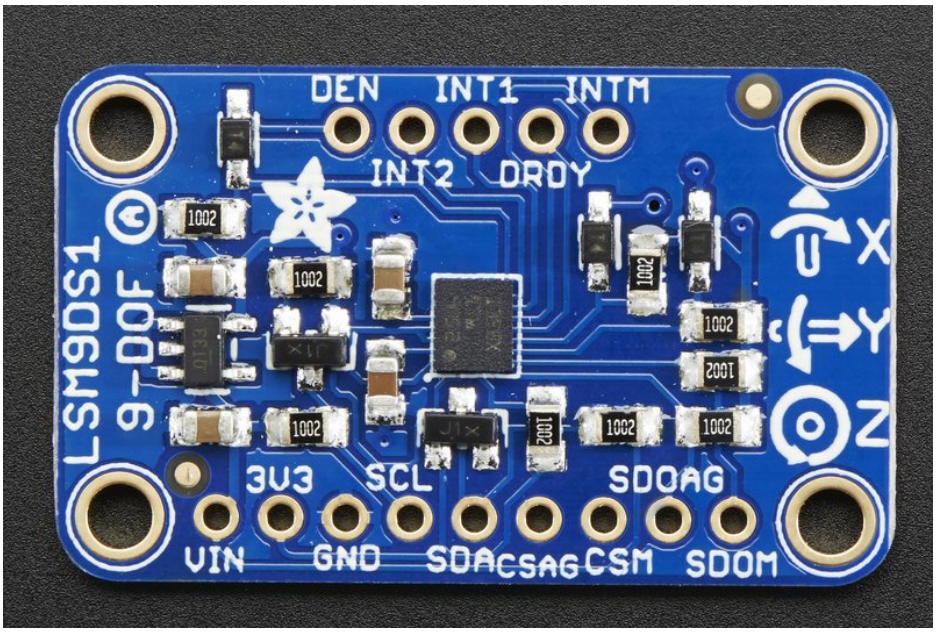
New New NEW for 2020!

The Classic Blue revision of the popular LSM9DS1 breakout has a new baby sister in Adafruit Black! Fabbed at 1 x 0.7 inches and weighing just a few grams, she's everything you like about the Classic Blue LSM9DS1 in a smaller package, and somehow we managed to *add* the new SparkFun qwiic (https://adafru.it/Fpw) compatible STEMMA QT (https://adafru.it/Ft4) connectors for the I2C bus so you don't even need to solder! Just wire one up to your favorite micro with a plug-and-play cable (https://adafru.it/JnB) and use our example code and libraries to measure those DoF's!

Pinouts


LSM9DS1 9DoF
Y
X
Z
SCL
A
M
VIN
3V
G
SDA
CS
SDO


DRDY
INT1
DEN
INTM
INT2
Gyro: ±245/500/2000dps
Mag: ±4/8/12/16 gauss
Accel: ±2/4/8/16g
I2C Addrs 0x1E,0x6B
VLogic/Vcc: 3-5VDC
SDOAG
CSAG
SCL
3Vo
DOM
CSM
SDA
GND
Vin



Power Pins

The sensor on the breakout requires 3V power. Since many customers have 5V microcontrollers like Arduino, we tossed a 3.3V regulator on the board. Its ultra-low dropout so you can power it from 3.3V-5V just fine.

- **Vin** - this is the power pin. Since the chip uses 3 VDC, we have included a voltage regulator on board that will take 3-5VDC and safely convert it down. To power the board, give it the same power as the logic level of your microcontroller - e.g. for a 5V micro like Arduino, use 5V
- **3V3** - this is the 3.3V output from the voltage regulator, you can grab up to 100mA from this if you like
- **GND** - common ground for power and logic

I2C Pins

- **SCL** - I2C clock pin, connect to your microcontrollers I2C clock line. This pin is level shifted so you can use 3-5V logic, and there's a 10K pullup on this pin.
- **SDA** - I2C data pin, connect to your microcontrollers I2C data line. This pin is level shifted so you can use 3-5V logic, and there's a 10K pullup on this pin.

See the back of the Adafruit Black version for the full pin labels.

SPI Pins

If you're interested in using SPI to interface with the LSM9DS1, you can!

- **SCL** - this is also the SPI clock pin, it's level shifted so you can use 3-5V logic input
- **SDA** - this is also the SPI MOSI pin, it's level shifted so you can use 3-5V logic input
- **CSAG** - this is the Accelerometer+Gyro subchip Chip Select, it's level shifted so you can use 3-5V logic input
- **CSM** - this is the Magnetometer subchip Select, it's level shifted so you can use 3-5V logic input
- **SDOAG** - this is the Accelerometer+Gyro subchip MISO pin - it's 3V logic out, but can be read properly by 5V logic chips.
- **SDOM/DOM**- this is the Magnetometer subchip MISO pin - it's 3V logic out, but can be read properly by 5V logic chips.

Interrupt & Misc Pins

Since there's so many sensors in the LSM9DS1, there's quite a number of interrupt outputs.

- **DEN** - this is a pin that supposedly could be used to dynamically enable/disable the Gyro. There's actually no documentation on it but we break it out for you anyways.
- **INT1** & **INT2** - These are interrupts from the accelerometer/gyro subchip. We don't have specific library support for these so check the datasheet for what you can make these indicate. They are 3V-logic outputs
- **DRDY** - this is the accelerometer/gyro subchip data ready output. We don't have specific library support for these so check the datasheet for how you can set the registers to enable this pin. It is a 3V-logic output.
- **INTM** - This is the interrupt from the magnetometer subchip. We don't have specific library support for it so check the datasheet for what you can make it indicate. It is a 3V-logic output.

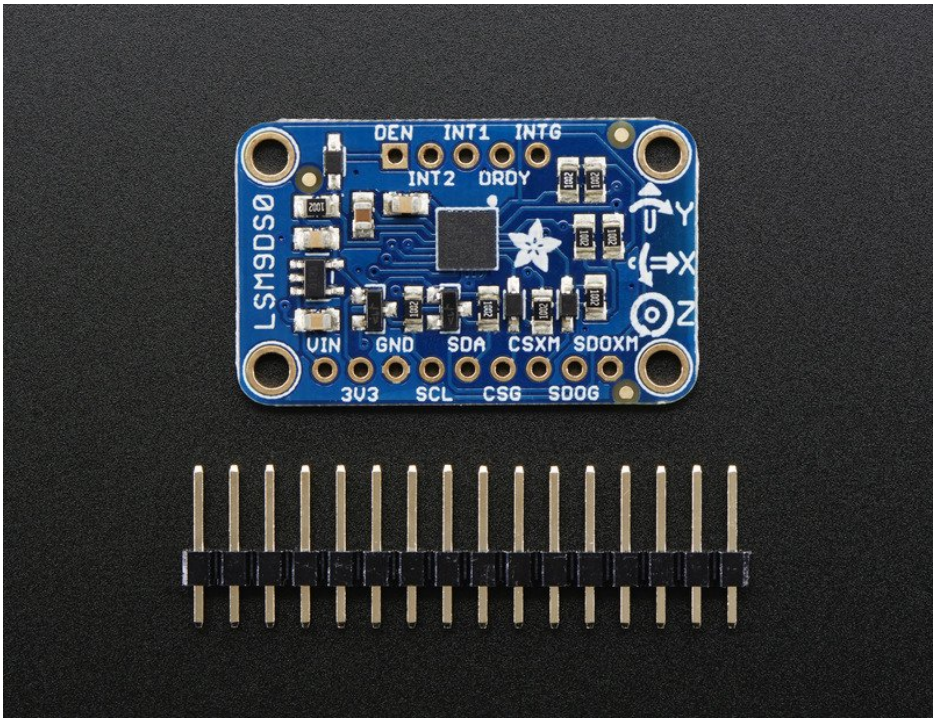
Note that the pinouts for the new Adafruit Black Stemma QT version are slightly different, with the positions of the DRDY and INTM pins swapped when compared to the Classic Blue revision

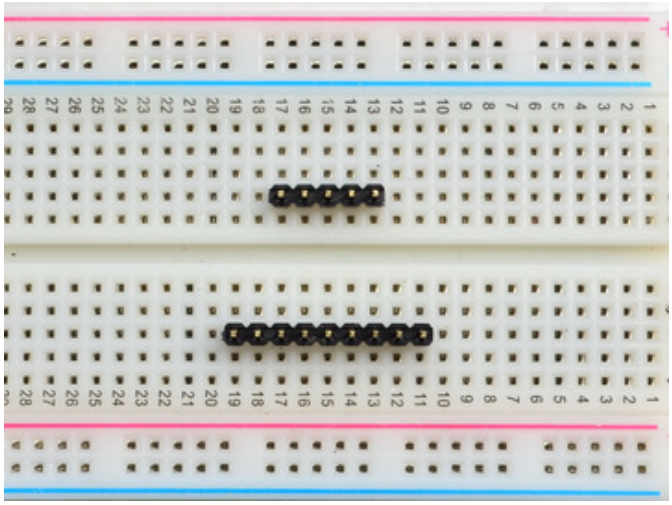
Black/Stemma QT Version only

- **STEMMA QT Connectors (https://adafru.it/Ft4) -** These connectors allow you to make solderless connections to dev boards and various other QT accessories (https://adafru.it/Ft6) using Stemma QT cables (https://adafru.it/JnB)

Assembly

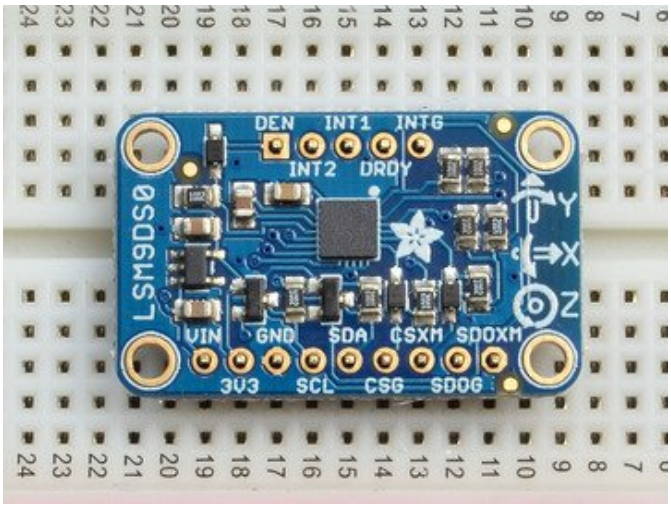
If you have the breadboard version of this sensor, you'll want to solder some header onto the sensor so it can be used in a breadboard. The Flora version does not require any extra assembly





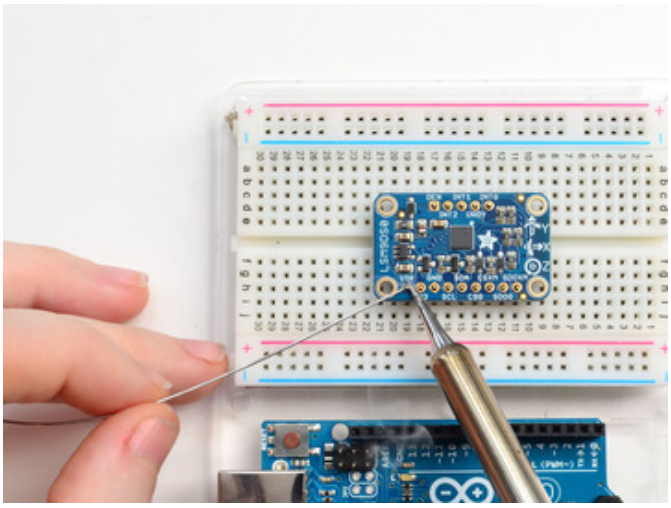
Prepare the header strip:

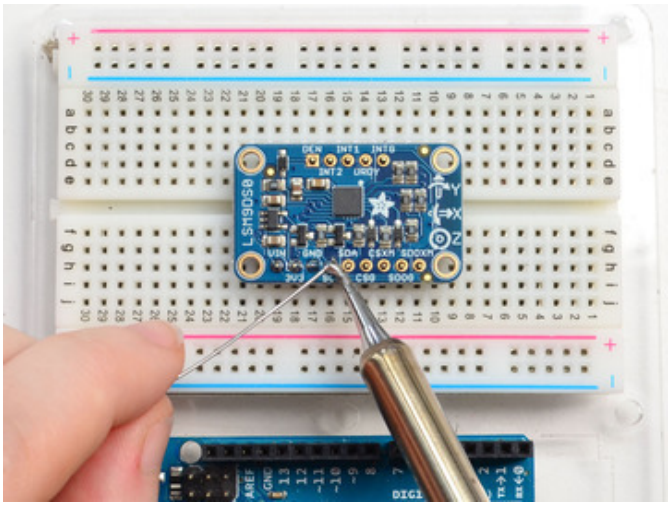
Cut the strip to length if necessary. It will be easier to solder if you insert it into a breadboard - **long pins down**



Add the breakout board:

Place the breakout board over the pins so that the short pins poke through the breakout pads



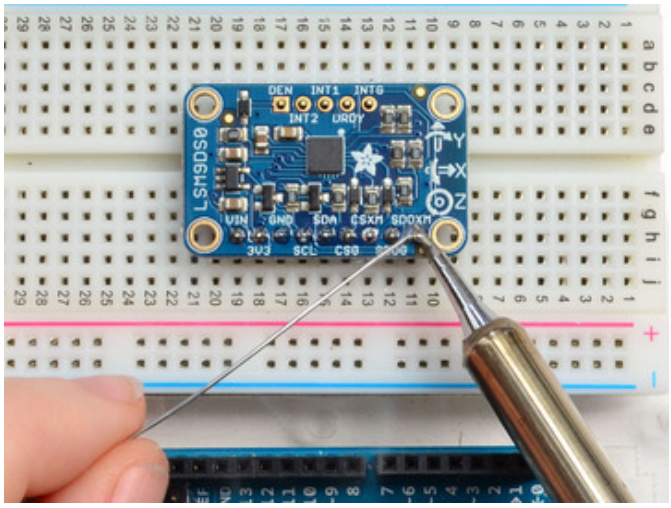


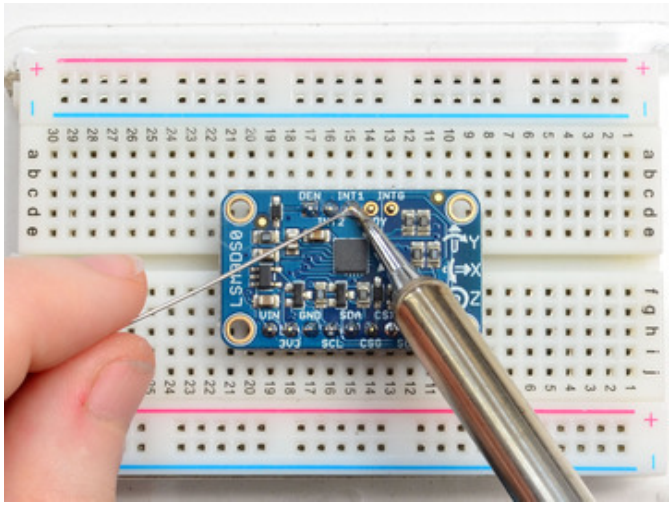
And Solder!

Be sure to solder all pins for reliable electrical contact.

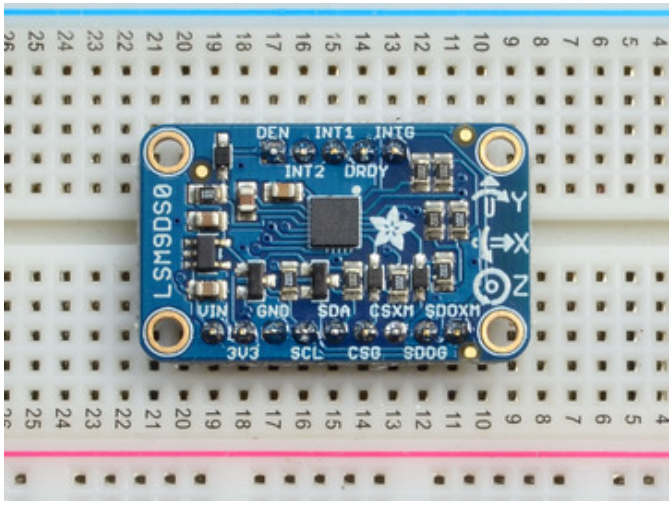
Solder the longer power/data strip first

(For tips on soldering, be sure to check out our Guide to Excellent Soldering (https://adafru.it/aTk).





If you plan to use the interrupts and/or you want the board to sit flatter in a breadboard, solder up the other strip!



You're done! Check your solder joints visually and continue onto the next steps

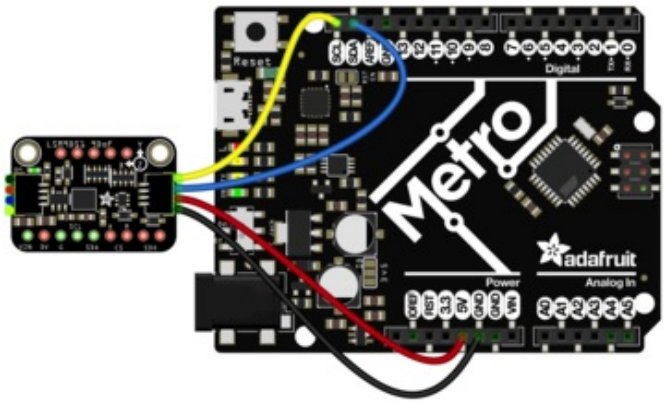
Arduino Code

Wiring for Arduino

Note that the new Adafruit black revision with Stemma QT connectors has some of the pin labels on the back of the board. See the Pinouts page for more information

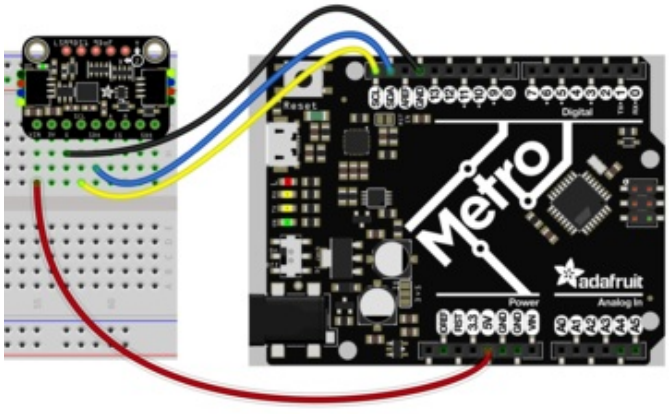
You can easily wire this breakout to any microcontroller, we'll be using a Metro, Adafruit's Arduino compatible board. For another kind of microcontroller, just make sure it has I2C or SPI, then port the code - its pretty simple stuff!

Let's start with just I2C interfacing since it requires the fewest number of wires. Here is how you would wire up one of the newer revision boards with the black PCB by using the new Stemma QT connectors:



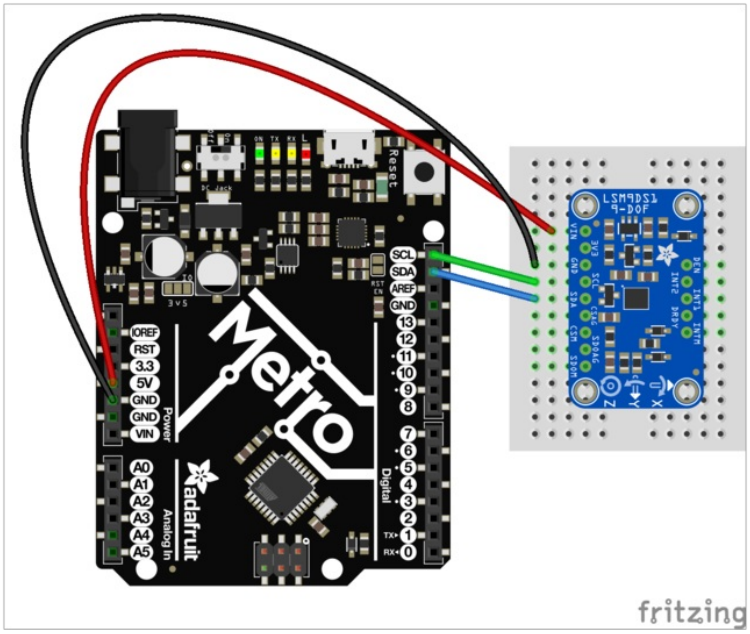
- Connect **board VIN (red wire)** to **Arduino 5V** if you are running a **5V** board Arduino (Uno, etc.). If your board is **3V,** connect to that instead.
- Connect **board GND (black wire)** to **Arduino GND**
- Connect **board SCL (yellow wire)** to **Arduino SCL**
- Connect **board SDA (blue wire)** to **Arduino SDA**

If you prefer to use a breadboard wire up the sensor like this:



- Connect **board VIN (red wire)** to **Arduino 5V** if you are running a **5V** board Arduino (Uno, etc.). If your board is **3V,** connect to that instead.
- Connect **board GND (black wire)** to **Arduino GND**
- Connect **board SCL (yellow wire)** to **Arduino SCL**
- Connect **board SDA (blue wire)** to **Arduino SDA**

If you have one of the classic blue versions of the LSM9DS1, you'll wire it up like so:

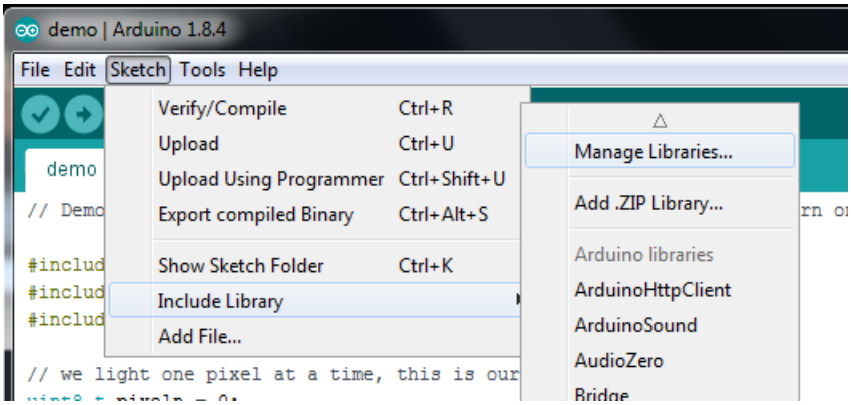


- Connect **Vin** to the power supply, 3-5V is fine. Use the same voltage that the microcontroller logic is based off of. For most Arduinos, that is 5V
- Connect **GND** to common power/data ground
- Connect the **SCL** pin to the I2C clock **SCL** pin on your Arduino. On an UNO & '328 based Arduino, this is also known as **A5**, on a Mega it is also known as **digital 21** and on a Leonardo/Micro, **digital 3**
- Connect the **SDA** pin to the I2C data **SDA** pin on your Arduino. On an UNO & '328 based Arduino, this is also known as **A4**, on a Mega it is also known as **digital 20** and on a Leonardo/Micro, **digital 2**

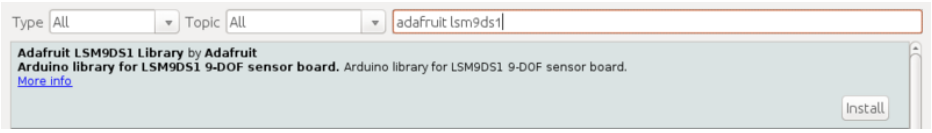
Download Arduino libraries

To begin reading sensor data, you will need to download the **Adafruit_LSM9DS1** library and the **Adafruit_Sensor** library from the Arduino library manager.

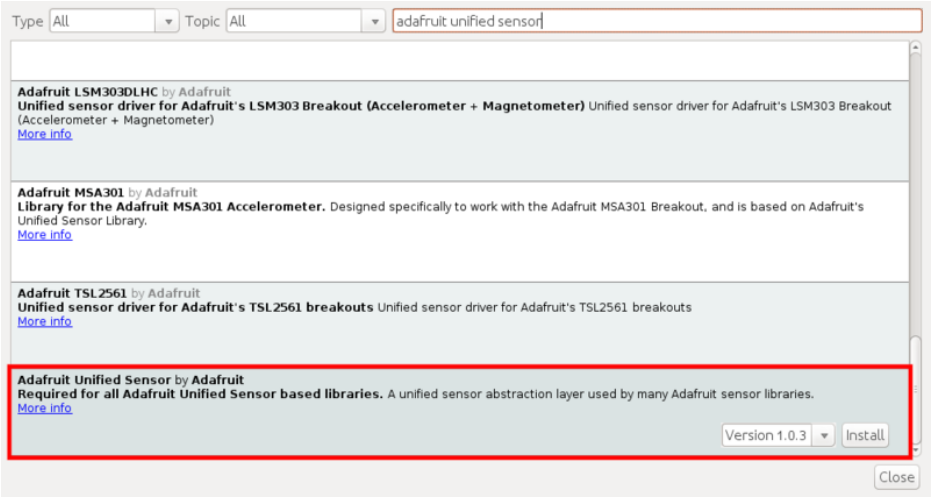
Open up the Arduino library manager:



Search for the **Adafruit LSM9DS1** library and install it



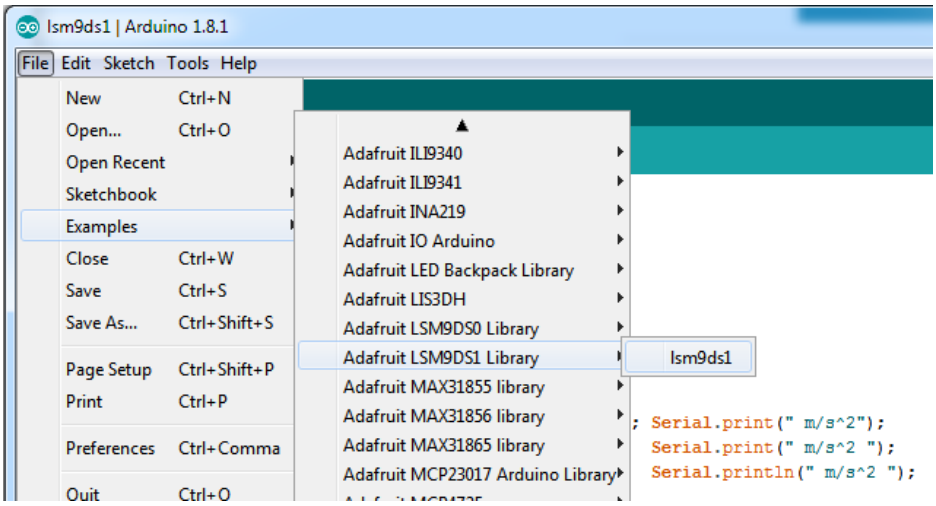
Search for the **Adafruit Unified Sensor** library and install it (You may have to scroll down pretty far to find it)



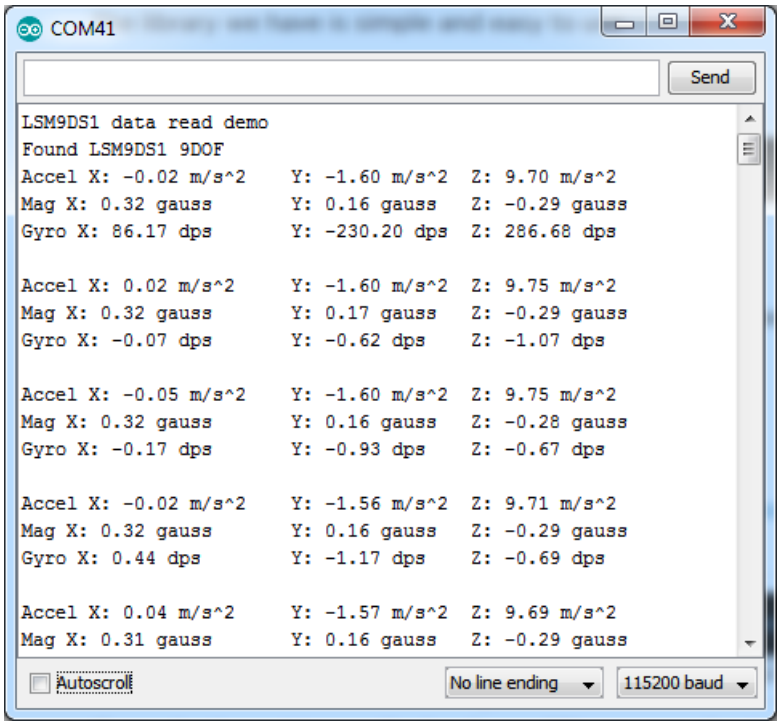
We also have a great tutorial on Arduino library installation at:
http://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use (https://adafru.it/aYM)

Load Demo Sketch

Now you can open up **File->Examples->Adafruit_LSM9DS1->lsm9ds1** and upload to your Arduino wired up to the sensor



Then open up the Serial console at 115200 baud to read the sensor output! You'll get 9 distinct data points, accelerometer x/y/z in meters/s^2, magetometer x/y/z in gauss and gyroscope in x/y/x degrees/second



We suggest using this **Adafruit_Sensor** interface as shown in this demo, since it will let you swap sensors without having to worry about units compatibility. Try twisting and moving the board around to see the sensors change value.

Library Reference

The library we have is simple and easy to use

You can create the **Adafruit_LSM9DS1** object with:

```
Adafruit_LSM9DS1 lsm = Adafruit_LSM9DS1();  // i2c sensor
```

I2C does not have pins, as they are fixed in hardware.

If you're using "hardware" SPI, you will have to wire up the pins as follows:

- **SCL** -> **SPI CLK**
- **SDA** -> **SPI MOSI**
- **SDO_AG** & **SDO_M** -> **SPI MISO** (both together)

You can determine the hardware SPI pins for your Arduino here (https://adafru.it/d5h) Then pick two pins for the CS lines

```
Adafruit_LSM9DS1 lsm = Adafruit_LSM9DS1(LSM9DS1_XGCS, LSM9DS1_MCS);
```

If you don't want to use the hardware SPI, you can also try the soft SPI capability, which is bitbanged. You can basically use any pins you like!

```
Adafruit_LSM9DS1 lsm = Adafruit_LSM9DS1(LSM9DS1_SCK, LSM9DS1_MISO, LSM9DS1_MOSI, LSM9DS1_XGCS,
LSM9DS1_MCS);
```

Begin!

To initialize the sensor, call **lsm.begin()** which will check the sensor can be found. It returns true/false depending on these checks. We suggest you wrap begin() in a statement that will check if the sensor was located:

```
if(!lsm.begin())
  {
    /* There was a problem detecting the LSM9DS1 ... check your connections */
    Serial.print(F("Ooops, no LSM9DS1 detected ... Check your wiring!"));
    while(1);
  }
```

Set Ranges

These chips have tons of registers, we basically provide interface code for the most useful stuff, such as setting the range. Each subsensor has it's own range. Higher ranges have less precision but can measure larger movements! Set up the ranges with the **setup** functions:

```
// 1.) Set the accelerometer range
  lsm.setupAccel(lsm.LSM9DS1_ACCELRANGE_2G);
  //lsm.setupAccel(lsm.LSM9DS1_ACCELRANGE_4G);
  //lsm.setupAccel(lsm.LSM9DS1_ACCELRANGE_8G);
  //lsm.setupAccel(lsm.LSM9DS1_ACCELRANGE_16G);

  // 2.) Set the magnetometer sensitivity
  lsm.setupMag(lsm.LSM9DS1_MAGGAIN_4GAUSS);
  //lsm.setupMag(lsm.LSM9DS1_MAGGAIN_8GAUSS);
  //lsm.setupMag(lsm.LSM9DS1_MAGGAIN_12GAUSS);
  //lsm.setupMag(lsm.LSM9DS1_MAGGAIN_16GAUSS);

  // 3.) Setup the gyroscope
  lsm.setupGyro(lsm.LSM9DS1_GYROSCALE_245DPS);
  //lsm.setupGyro(lsm.LSM9DS1_GYROSCALE_500DPS);
  //lsm.setupGyro(lsm.LSM9DS1_GYROSCALE_2000DPS);
```

Choose whichever range you like, after you begin() the sensor!

Read data

Read data using the Adafruit_Sensor API by first creating four events, one for each sub-sensor:

```
sensors_event_t accel, mag, gyro, temp;
```

Then pass these into the **getEvent** function

```
lsm.getEvent(&accel, &mag, &gyro, &temp);
```

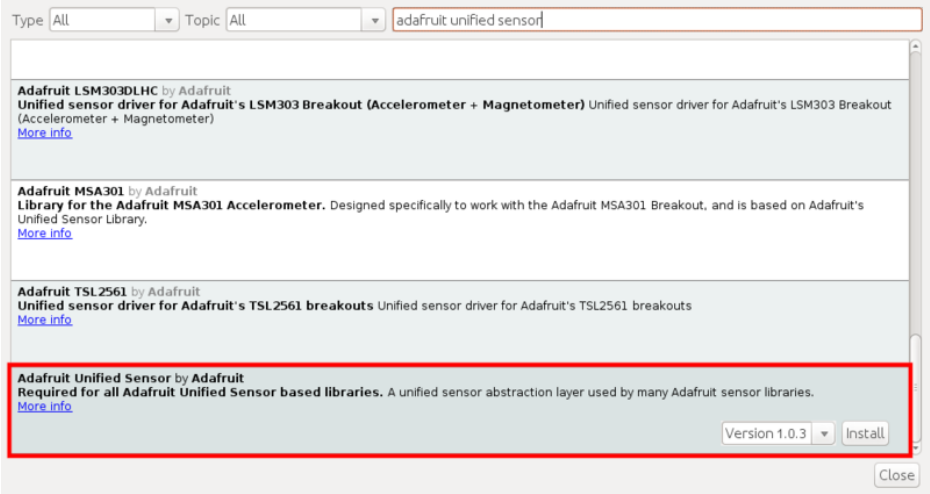
The data is snapshotted at once, so you can read and manage the data later.

For the Accelerometer event you can read **accel.acceleration.x**, **accel.acceleration.y** or **accel.acceleration.z** which are in meters/second*second.

For the Magnetometer event you can read **mag.magnetic.x**, **mag.magnetic.y** or **mag.magnetic.z** which are in gauss.

For the Gyro event you can read **gyro.gyro.x, gyro.gyro.y** or **gyro.gyro.z**, which are in degrees-per-second (dps)

The temperature event data is in **temp.temperature**, but we don't guarantee that the temperature data is in degrees C


Type All
Topic All
adafruit unified sensor
Adafruit LSM303DLHC by Adafruit
Unified sensor driver for Adafruit's LSM303 Breakout (Accelerometer + Magnetometer) Unified sensor driver for Adafruit's LSM303 Breakout (Accelerometer + Magnetometer)
More info
Adafruit MSA301 by Adafruit
Library for the Adafruit MSA301 Accelerometer. Designed specifically to work with the Adafruit MSA301 Breakout, and is based on Adafruit's Unified Sensor Library.
More info
Adafruit TSL2561 by Adafruit
Unified sensor driver for Adafruit's TSL2561 breakouts Unified sensor driver for Adafruit's TSL2561 breakouts
More info
Adafruit Unified Sensor by Adafruit
Required for all Adafruit Unified Sensor based libraries. A unified sensor abstraction layer used by many Adafruit sensor libraries.
More info
Version 1.0.3
Install
Close

Note that the new Adafruit black revision with Stemma QT connectors has some of the pin labels on the back of the board. See the Pinouts page for more information

CircuitPython Microcontroller Wiring

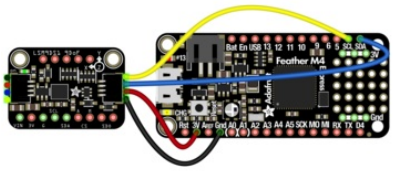
First wire up a LSM9DS1 to your board exactly as shown on the previous pages for Arduino. You can use either I2C or SPI wiring, although it's recommended to use I2C for simplicity.

I2C

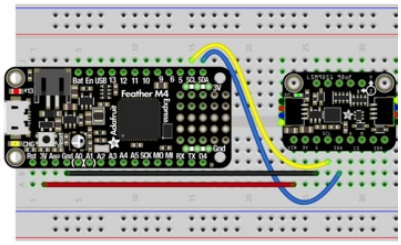
Here's an example of wiring a Feather M0 to the sensor with an I2C connection, either the new classic blue form factor, or the newer revisions either on a breadboard or using the Stemma QT connectors:

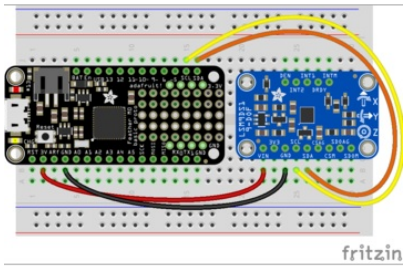
It's easy to use the LSM9DS1 sensor with Python or CircuitPython, and the Adafruit CircuitPython LSM9DS1 (https://adafru.it/C5b) module. This module allows you to easily write Python code that reads the accelerometer, magnetometer, and gyroscope from the sensor.

You can use this sensor with any CircuitPython microcontroller board or with a computer that has GPIO and Python thanks to Adafruit_Blinka, our CircuitPython-for-Python compatibility library (https://adafru.it/BSN).



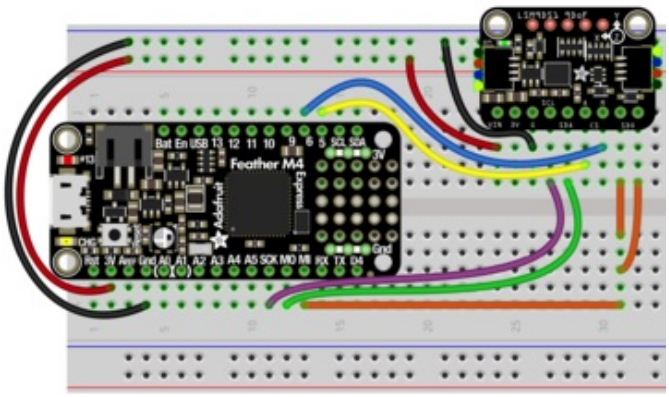
- **Board 3V** to **sensor VIN**
- **Board GND** to **sensor GND**
- **Board SCL** to **sensor SCL**
- **Board SDA** to **sensor SDA**





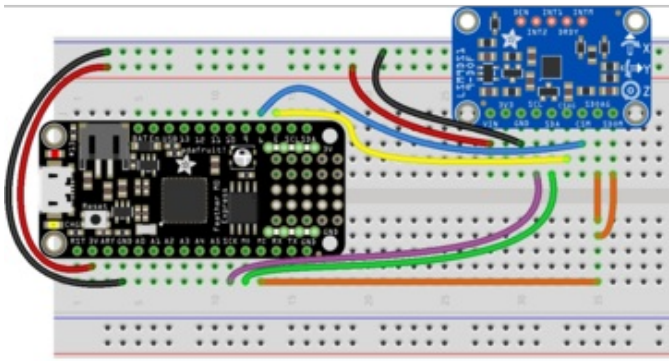
SPI

Wiring up the LSM9DS1 to a Feather M0 using SPI is a bit more complicated due to the addition of two sets of Sensor Out and CS pins. Wiring for the newer or older versions of the board are pretty similar



- **Board 3V** to **sensor VIN (Red)**
- **Board GND** to **sensor GND (Black)**
- **Board SCK** to **sensor SCL (Purple)**
- **Board MOSI** to **sensor SDA (Green)**
- **Board MISO** to **sensor SDOAG** AND **sensor DOM (Orange)**
- **Board D5** to **sensor CSAG (Yellow)**
- **Board D6** to **sensor CSM (Blue)**

To use the classic blue board with SPI

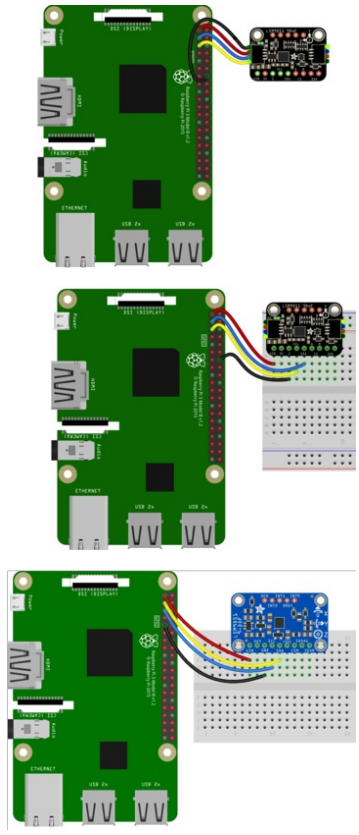


- **Board 3V** to **sensor VIN (Red)**
- **Board GND** to **sensor GND (Black)**
- **Board SCK** to **sensor SCL (Purple)**
- **Board MOSI** to **sensor SDA (Green)**
- **Board MISO** to **sensor SDOAG** AND **sensor SDOM (Orange)**
- **Board D5** to **sensor CSAG (Yellow)**
- **Board D6** to **sensor CSM (Blue)**

Python Computer Wiring

Since there's *dozens* of Linux computers/boards you can use we will show wiring for Raspberry Pi. For other platforms, please visit the guide for CircuitPython on Linux to see whether your platform is supported (https://adafru.it/BSN).

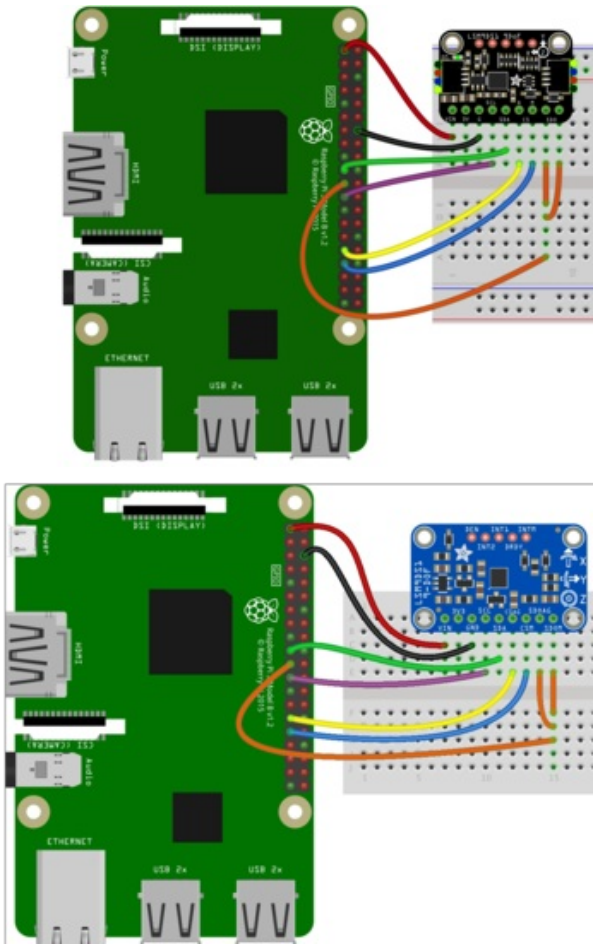
I2C



- **Pi 3V3** to **sensor VIN**
- **Pi GND** to **sensor GND**
- **Pi SCL** to **sensor SCL**
- **Pi SDA** to **sensor SDA**

SPI

Here's the Raspberry Pi wired with SPI:



- **Pi 3V3** to **sensor VIN**
- **Pi GND** to **sensor GND**
- **Pi SCLK** to **sensor SCL**
- **Pi MOSI** to **sensor SDA**
- **Pi MISO** to **sensor SDOAG** AND **sensor SDOM**
- **Pi GPIO5** to **sensor CSAG**
- **Pi GPIO6** to **sensor CSM**

CircuitPython Installation of LSM9DS1 Library

You'll need to install the Adafruit CircuitPython LSM9DS1 (https://adafru.it/C5b) library on your CircuitPython board.

First make sure you are running the latest version of Adafruit CircuitPython (https://adafru.it/Amd) for your board.

Next you'll need to install the necessary libraries to use the hardware--carefully follow the steps to find and install these libraries from Adafruit's CircuitPython library bundle (https://adafru.it/zdx). Our introduction guide has a great page on how to install the library bundle (https://adafru.it/ABU) for both express and non-express boards.

Remember for non-express boards like the, you'll need to manually install the necessary libraries from the bundle:

- **adafruit_lsm9ds1.mpy**
- **adafruit_bus_device**

Before continuing make sure your board's lib folder or root filesystem has the **adafruit_lsm9ds1.mpy,** and **adafruit_bus_device** files and folders copied over.

Next connect to the board's serial REPL (https://adafru.it/Awz) so you are at the CircuitPython >>> prompt.

CircuitPython & Python Usage

To demonstrate the usage of the sensor we'll initialize it and read the accelerometer, magnetometer, and more from the board's Python REPL.

Run the following code to import the necessary modules and initialize the I2C connection with the sensor:

```
import board
import busio
import adafruit_lsm9ds1
i2c = busio.I2C(board.SCL, board.SDA)
sensor = adafruit_lsm9ds1.LSM9DS1_I2C(i2c)
```

If you're connected using SPI, run the following code to initialise the SPI connection with the sensor:

```
import board
import busio
from digitalio import DigitalInOut, Direction
spi = busio.SPI(board.SCK, board.MOSI, board.MISO)
csag = DigitalInOut(board.D5)
csag.direction = Direction.OUTPUT
csag.value = True
csm = DigitalInOut(board.D6)
csm.direction = Direction.OUTPUT
csm.value = True
sensor = adafruit_lsm9ds1.LSM9DS1_SPI(spi, csag, csm)
```

Now you're ready to read values from the sensor using any of these properties:

- **acceleromter** - A 3-tuple of X, Y, Z axis accelerometer values in meters/second squared.
- **magnetometer** - A 3-tuple of X, Y, Z axis magnetometer values in gauss.
- **gyroscope** - A 3-tuple of X, Y, Z axis gyroscope values in degrees/second.
- **temperature** - The sensor temperature in degrees Celsius.

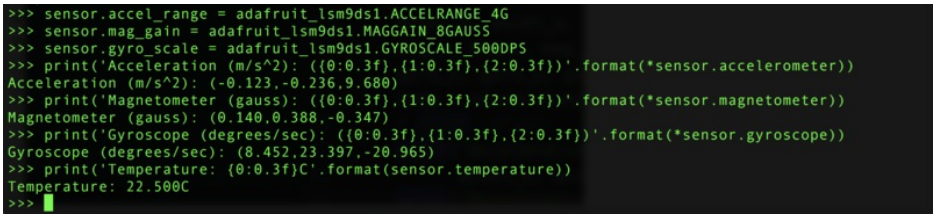
```
print('Acceleration (m/s^2): ({0:0.3f},{1:0.3f},{2:0.3f})'.format(*sensor.accelerometer))
print('Magnetometer (gauss): ({0:0.3f},{1:0.3f},{2:0.3f})'.format(*sensor.magnetometer))
print('Gyroscope (degrees/sec): ({0:0.3f},{1:0.3f},{2:0.3f})'.format(*sensor.gyroscope))
print('Temperature: {0:0.3f}C'.format(sensor.temperature))
```

```
>>> print('Acceleration (m/s^2): ({0:0.3f},{1:0.3f},{2:0.3f})'.format(*sensor.accelerometer))
Acceleration (m/s^2): (-0.121,-0.246,9.721)
>>> print('Magnetometer (gauss): ({0:0.3f},{1:0.3f},{2:0.3f})'.format(*sensor.magnetometer))
Magnetometer (gauss): (0.132,0.374,-0.331)
>>> print('Gyroscope (degrees/sec): ({0:0.3f},{1:0.3f},{2:0.3f})'.format(*sensor.gyroscope))
Gyroscope (degrees/sec): (8.356,23.144,-21.079)
>>> print('Temperature: {0:0.3f}C'.format(sensor.temperature))
Temperature: 22.500C
>>>
```

In addition you can adjust some properties by getting and setting their values:

- **accel_range** - The range of the accelerometer, should be a value of ACCELRANGE_2G, ACCELRANGE_4G, ACCELRANGE_8G, or ACCELRANGE_16G from the adafruit_lsm9ds1 module. The default is 2G.
- **mag_gain** - The gain of the magnetometer, should be a value of MAGGAIN_4GAUSS, MAGGAIN_8GAUSS, MAGGAIN_12GAUSS, or MAGGAIN_16GAUSS from the adafruit_lsm9ds1 module. The default is 4 gauss.
- **gyro_scale** - The scale of the gyroscope, should be a value of GYROSCALE_245DPS, GYROSCALE_500DPS, or GYROSCALE_2000DPS from the adafruit_lsm9ds1 module. The default is 245 DPS.

```
sensor.accel_range = adafruit_lsm9ds1.ACCELRANGE_4G
sensor.mag_gain = adafruit_lsm9ds1.MAGGAIN_8GAUSS
sensor.gyro_scale = adafruit_lsm9ds1.GYROSCALE_500DPS
```



See the simpletest.py example (https://adafru.it/C5d) for a complete demo of printing the accelerometer, magnetometer, gyroscope every second. Save this as **code.py** on the board and examine the REPL output to see the range printed every second.

That's all there is to using the LSM9DS1 with CircuitPython!

Full Example Code

```
# Simple demo of the LSM9DS1 accelerometer, magnetometer, gyroscope.
# Will print the acceleration, magnetometer, and gyroscope values every second.
import time
import board
import busio
import adafruit_lsm9ds1

# I2C connection:
i2c = busio.I2C(board.SCL, board.SDA)
sensor = adafruit_lsm9ds1.LSM9DS1_I2C(i2c)

# SPI connection:
# from digitalio import DigitalInOut, Direction
# spi = busio.SPI(board.SCK, board.MOSI, board.MISO)
# csag = DigitalInOut(board.D5)
# csag.direction = Direction.OUTPUT
# csag.value = True
# csm = DigitalInOut(board.D6)
# csm.direction = Direction.OUTPUT
# csm.value = True
# sensor = adafruit_lsm9ds1.LSM9DS1_SPI(spi, csag, csm)

# Main loop will read the acceleration, magnetometer, gyroscope, Temperature
# values every second and print them out.
while True:
    # Read acceleration, magnetometer, gyroscope, temperature.
    accel_x, accel_y, accel_z = sensor.acceleration
    mag_x, mag_y, mag_z = sensor.magnetic
    gyro_x, gyro_y, gyro_z = sensor.gyro
    temp = sensor.temperature
    # Print values.
    print(
        "Acceleration (m/s^2): ({0:0.3f},{1:0.3f},{2:0.3f})".format(
            accel_x, accel_y, accel_z
        )
    )
    print(
        "Magnetometer (gauss): ({0:0.3f},{1:0.3f},{2:0.3f})".format(mag_x, mag_y, mag_z)
    )
    print(
        "Gyroscope (degrees/sec): ({0:0.3f},{1:0.3f},{2:0.3f})".format(
            gyro_x, gyro_y, gyro_z
        )
    )
    print("Temperature: {0:0.3f}C".format(temp))
    # Delay for a second.
    time.sleep(1.0)
```

Python Docs

Python Docs (https://adafru.it/C5e)

Downloads

Files

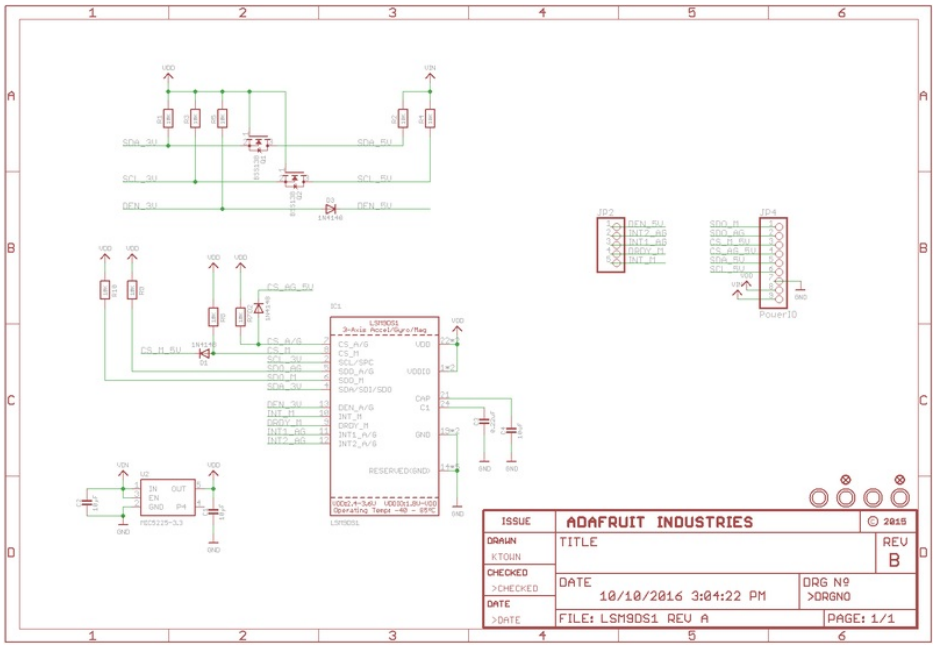
- EagleCAD PCB files on GitHub (https://adafru.it/ub2)
- Classic/Blue Fritzing object available in Adafruit Fritzing library (https://adafru.it/LKA)
- Stemma QT/Black Fritzing object available in Adafruit Fritzing library (https://adafru.it/LKB)

https://adafru.it/ub3

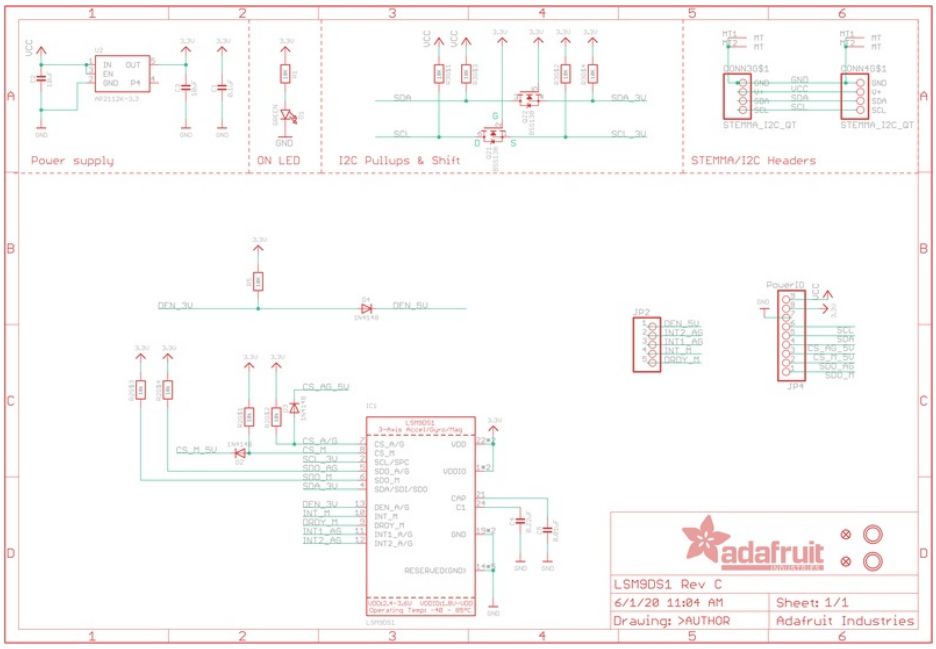
https://adafru.it/ub3

Schematic

Classic/Blue

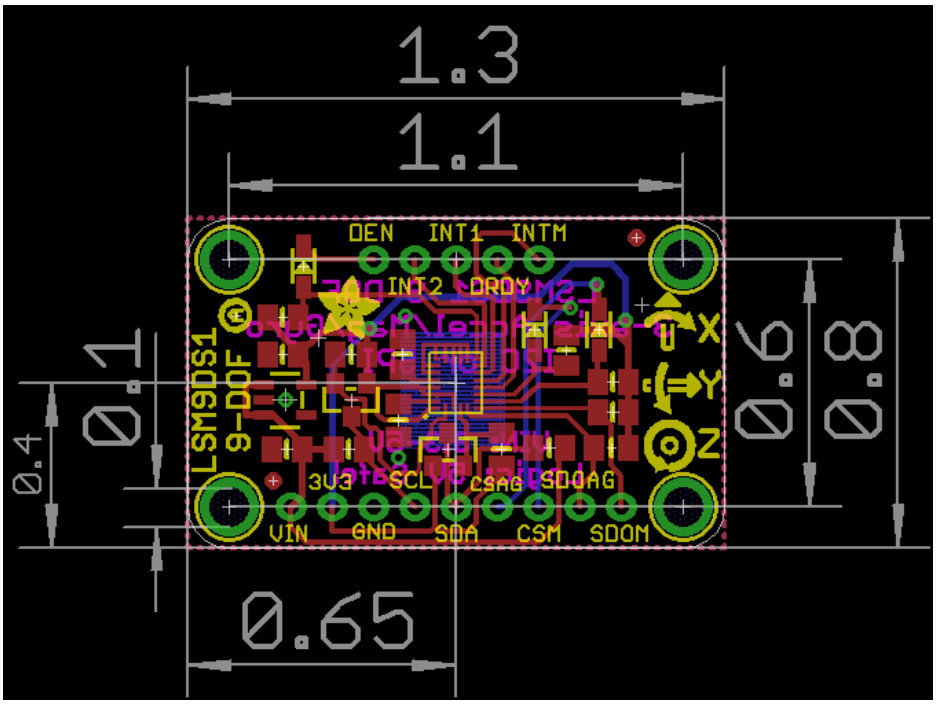


Stemma QT/Black

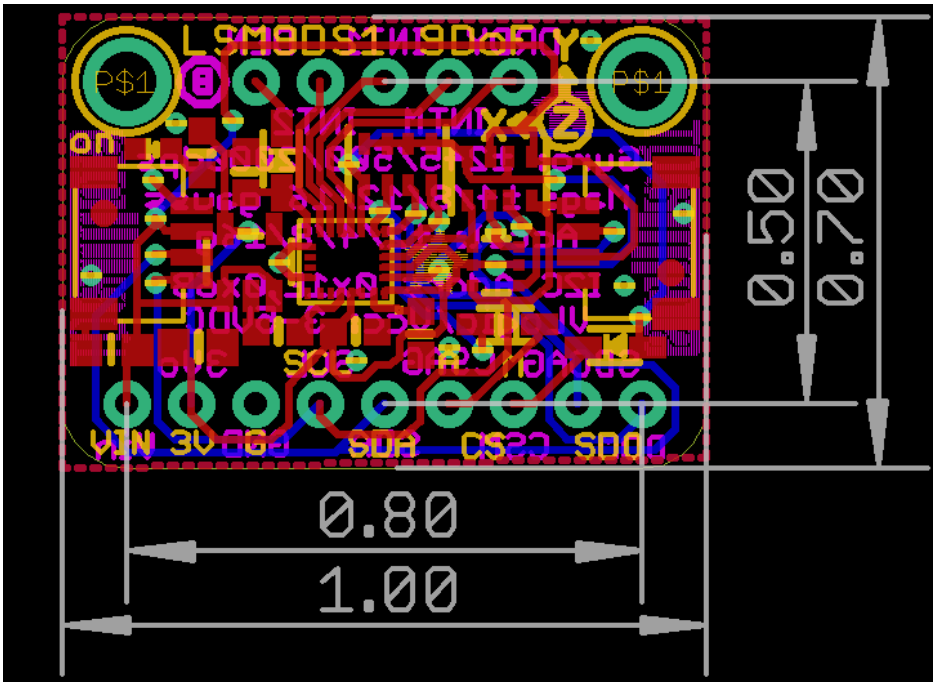


Fabrication Print

Classic/Blue



Stemma QT/Black


0.50
0.70
0.80
1.00